**Libreraium**

1. **Introducción**

Libreraium es un sistema de e-commerce basado en microservicios para la compra de libros digitales y físicos, accesible mediante una interfaz de consola. El sistema permite a los usuarios realizar búsquedas de productos, gestionarlos en el carrito y procesar compras, todo a través de un cliente basado en consola.

* 1. **Objetivos del Proyecto**
* Desarrollar una aplicación distribuida con una arquitectura de microservicios utilizando tecnologías como gRPC y MOM.
* Implementar un API Gateway en Java que maneje la autenticación de usuarios.
* Crear microservicios en Node.js y C++ para la gestión de productos, carrito y pedidos.
* Garantizar la tolerancia a fallos mediante el uso de un Middleware MOM en C++ con Rabbit  
  1. **Alcance**
* El sistema no tiene UI web, y funciona exclusivamente a través de una interfaz de consola.
* El API Gateway centraliza las solicitudes del cliente y las distribuye entre los microservicios.
* El sistema consta de tres microservicios: Usuarios, Productos (Libros), y Carrito/Pedidos.
* Se implementa un Middleware MOM para manejar la comunicación asíncrona en caso de fallos y asegurar la persistencia de los mensajes.
* El sistema genera archivos de log para registrar eventos y errores.

1. **Requerimientos del sistema**
   1. **Requerimientos Funcionales**

* Autenticación de usuarios: El sistema debe permitir el registro e inicio de sesión de los usuarios.
* Gestión de productos: El microservicio de productos debe permitir buscar, agregar, actualizar y eliminar libros disponibles.
* Carrito de compras: Los usuarios deben poder agregar o eliminar libros del carrito de compras.
* Procesamiento de pedidos: Los usuarios deben poder procesar compras y realizar pagos de forma segura.
* Manejo de fallos: El sistema debe garantizar la entrega de mensajes a través de RabbitMQ/Kafka en caso de fallos.
  1. **Requerimientos No Funcionales**
* Escalabilidad: El sistema debe ser capaz de manejar un incremento de usuarios y servicios de manera eficiente.
* Seguridad: El sistema debe tener mecanismos de autenticación con JWT y autorización para asegurar que los usuarios solo tengan acceso a los recursos permitidos.
* Tolerancia a fallos: El sistema debe ser resistente a caídas, utilizando el Middleware MOM para garantizar la entrega de mensajes incluso cuando un microservicio no esté disponible.
* Rendimiento: La latencia de la comunicación entre los microservicios debe ser mínima, con tiempos de respuesta rápidos.

1. **Modelo de Arquitectura**
   1. **C4**
      1. **Diagrama de Contexto**

[**https://lucid.app/lucidchart/67e71b2c-10c1-4953-87b5-a9fed9e28d41/edit?invitationId=inv\_c07f0cfd-4002-468e-84a8-14132aff1080&page=b9S2d\_7k1Kx\_#**](https://lucid.app/lucidchart/67e71b2c-10c1-4953-87b5-a9fed9e28d41/edit?invitationId=inv_c07f0cfd-4002-468e-84a8-14132aff1080&page=b9S2d_7k1Kx_#)

* + 1. **Diagrama de Contenedores**
    2. **Diagrama de Componentes**
  1. **UML**
     1. **Diagrama de Clases**
     2. **Diagrama de Secuencia**
     3. **Diagrama de Comunicación**

1. **Entregables**

* Código Fuente del API Gateway y microservicios.
* Documentación técnica detallada de la arquitectura, diagramas y componentes.
* Pruebas de integración que validan la comunicación entre los microservicios.
* Logs de eventos y errores generados por el sistema durante la ejecución.
* Presentación final del proyecto.

1. **Conclusión**

Libreraium es un sistema robusto de e-commerce basado en microservicios que emplea gRPC, JWT y RabbitMQ/Kafka para asegurar la tolerancia a fallos y garantizar una comunicación eficiente entre los componentes del sistema. La utilización de un API Gateway en Java y microservicios en Node.js y C++ permite un desarrollo modular y escalable, mientras que el Middleware MOM asegura la entrega de mensajes de manera confiable. Este proyecto proporciona una base sólida para la implementación de aplicaciones distribuidas modernas con una arquitectura de microservicios.